LINMA2300: Project 1
Dimensionality reduction methods and nearest neighbor problem

Prof. Laurent Jacques, Bastien Massion, Nicolas Mil-Homens Cavaco

21th October 2024, v2

1 Context

Consider we have a dataset $\boldsymbol{X} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_N] \in \mathbb{R}^{n \times N}$ containing $N > 0$ images $\boldsymbol{x}_i \in \mathbb{R}^n$ of $n = 3p^2 > 0$ pixels flattened in columns (3 color channels for $p \times p$ images). Given a new image $\boldsymbol{q} \in \mathbb{R}^n$ not contained in the dataset, we would like to find the closest data point from $\boldsymbol{q}$ in $\{\boldsymbol{x}_i\}_{i=1}^N$. In other terms, we would like to find

$$\hat{\boldsymbol{q}} := \boldsymbol{x}_{i^*} \in \mathbb{R}^n \text{ such that } i^* := \arg\min_{i=1,\dots,N} \|\boldsymbol{x}_i - \boldsymbol{q}\|_2.$$

This might have a lot of applications, such as classification or clustering (but we are not concerned about any particular future application for this project). However, in the case of a high dimensional dataset containing images with a large number of pixels, finding $\hat{\boldsymbol{q}}$ might be computationally expensive since it requires the computation of the distances between $\boldsymbol{q}$ and every $\boldsymbol{x}_i$ in the dataset.

In this project, we would like to reduce the complexity of computing $\hat{\boldsymbol{q}}$ and propose to reduce the dimension n to d such that $0 < d \ll n$ by projecting the original data points on the column space of a certain matrix $\boldsymbol{\Phi} \in \mathbb{R}^{d \times n}$. The projections are denoted $\boldsymbol{x}'_i \in \mathbb{R}^d$, for $1 \leqslant i \leqslant N$. More concretely, we proceed in two steps:

Projected nearest neighbor method:

1. For a given projection matrix $\boldsymbol{\Phi} \in \mathbb{R}^{d \times n}$, define $\boldsymbol{X}' = \boldsymbol{\Phi}\boldsymbol{X} \in \mathbb{R}^{d \times N}$

2. For a given $\boldsymbol{q} \in \mathbb{R}^n$, compute $\boldsymbol{q}' = \boldsymbol{\Phi}\boldsymbol{q} \in \mathbb{R}^d$ and

$$\hat{\boldsymbol{q}} := \boldsymbol{x}_{i^*} \in \mathbb{R}^n \text{ such that } i^* := \arg\min_{i=1,\ldots,N} \|\boldsymbol{x}'_i - \boldsymbol{q}'\|_2,$$

where $\boldsymbol{X}' = [\boldsymbol{x}'_1 \cdots \boldsymbol{x}'_N] \in \mathbb{R}^{d \times N}$.

2 Dataset

For this project, we will use the dataset CIFAR-10: $p = 32$, $n = 3072$, $N = 8000$. It is available as a zip file at `https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz`. You should unzip the file in the same working directory as the provided Notebook.

In the Notebook, the dataset is loaded and split into two parts: the dataset $\boldsymbol{X}$ and the test set containing 100 additional queries $\boldsymbol{q}$. In reality, the complete CIFAR-10 dataset contains $N = 50000$ images and 10000 additional queries: you can use it if you want, but sticking to the numbers cited above is sufficient. Note that we do not care about the image labels for this project.

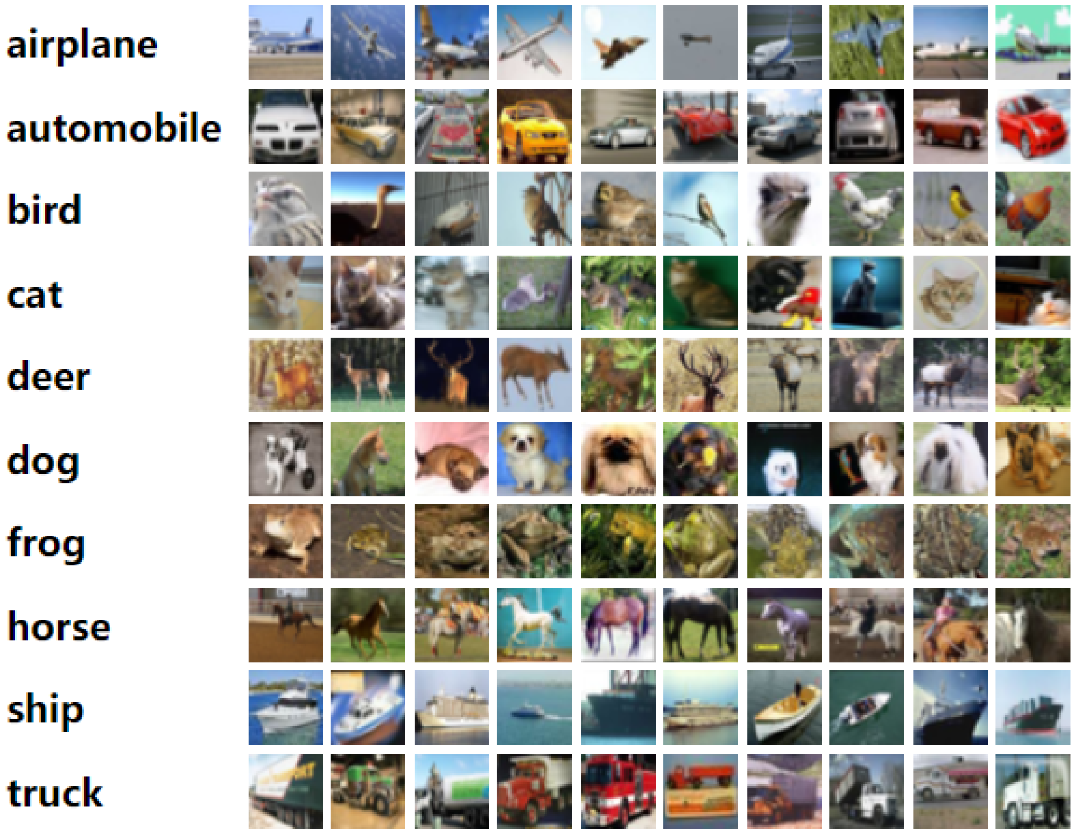

Figure 1: Sample of CIFAR-10

3 Project

3.1 Identity projection

1. First consider $d = n$ and $\boldsymbol{\Phi} = \boldsymbol{I}_n$, which is the $n \times n$ identity matrix. The projection does nothing, i.e., it is equivalent to solving the initial problem. Implement the "projected nearest neighbor" algorithm using the identity matrix and run it on the data set $\boldsymbol{X}$ for each image $\boldsymbol{q}$ of the test set.

   Pkoi faire ça et pas juste prendre le X de base ?

2. Compute the theoretical time complexity with respect to N and n.

   Pas de valeur numérique ? Ou alors on
   donne une formule de la time complexity
   en fct de N et n puis une valeur numérique
   en remplacant N et n par leur valeur ?

3.2 PCA projection

Now we are interested by reducing the dimension n by projecting the data set on the $d < n$ first principal axis of $\boldsymbol{X}$ found by PCA. To do so, one can rely on the SVD of the dataset $\boldsymbol{X} = \boldsymbol{U\Sigma V}^\top$ and define $\boldsymbol{\Phi} = \boldsymbol{U}_d^\top$, where $\boldsymbol{U}_d$ contains the d first left singular vectors of $\boldsymbol{X}$.

1. Compute the matrix $\boldsymbol{\Phi} = \boldsymbol{U}_d^\top$ and apply the "projected nearest neighbor" with $d = 300$. Compare your result with the one in question 3.1.

   comment comparer explicitment ? Packe bcp d'images pas possible de faire l
   analyse one by one

   mtn j'ai d =300
   avant j'avais 3072 donc p*p*3 = 3072 avec p=32 ok
   mtn je peux plus utiliser p=32. comment faire ? Je peux faire sqrt(d/3)? J'ai
   essayé d'imposer p=10 mais ca fonctionne pas ....

2. What is the time complexity of your algorithm theoretically? Compare your formula with the empirical time complexity through numerical experiments by varying d.

   Hint: Use log-log plots to represent time-complexity.

3. How does the error evolve with d empirically? In particular, plot how the accuracy evolves, i.e. the number of correctly recovered neighbors divided by the number of data points in the test set, with respect to the ratio $\frac{d}{n}$.

3.3 Gaussian projection

We now propose to use a Gaussian random matrix $\boldsymbol{\Phi} \in \mathbb{R}^{d \times n}$ with i.i.d. entries following $\mathcal{N}(\mu = 0, \sigma^2 = \frac{1}{d})$.

1. Why is it a good idea to rely on a Gaussian random matrix $\boldsymbol{\Phi}$ for our specific dataset $\boldsymbol{X}$ containing images?

   Hint: it is assumed that an image is approximately sparse in a certain basis of functions called wavelets. Mathematically, for any image $\boldsymbol{x} \in \mathbb{R}^n$ of the dataset, one can find an approximately sparse vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ such that $\boldsymbol{x} = \boldsymbol{\Psi\alpha} = \sum_{j=1}^n \alpha_j \boldsymbol{\psi}_j$, where $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1 \cdots \boldsymbol{\psi}_n] \in \mathbb{R}^{n \times n}$ and

$\boldsymbol{\psi}_j \in \mathbb{R}^n$ are real wavelet basis functions. Moreover, you can assume that the wavelet basis is orthonormal: $\boldsymbol{\Psi}^\top \boldsymbol{\Psi} = \boldsymbol{I}_n$.

Hint 2: If some matrix $\boldsymbol{A}$ is defined as $\boldsymbol{A} = \boldsymbol{\Phi}\boldsymbol{\Psi}$, how are its entries distributed?

2. Implement and apply the "projected nearest neighbor" with this random matrix $\boldsymbol{\Phi}$ and $d = 300$.

3. What is the time complexity of your algorithm theoretically? Compare your formula with the empirical time complexity through numerical experiments by varying d.

4. Theoretically, what can you say about the error made by your random projection method compared with the solution obtained in question 3.1? Numerically, how does the error evolve with respect to d? Again, plot how the accuracy evolves with respect to the ratio $\frac{d}{n}$. Compare your results with the one obtained with PCA in question 3.2.

Hint. Don't forget that your matrix $\boldsymbol{\Phi}$ is random: average over several runs to get consistent results.

3.4 A particular random projection

Finally, consider the matrix $\boldsymbol{\Phi} = \boldsymbol{S}_\Gamma \boldsymbol{F}^* \boldsymbol{D} \boldsymbol{F}$ where $\boldsymbol{F} \in \mathbb{C}^{n \times n}$ is the DFT matrix and $\boldsymbol{S}_\Gamma \in \mathbb{R}^{d \times n}$ is a matrix that randomly selects d rows indexed by the subset $\Gamma \subset [n]$, such that $|\Gamma| = d$. The matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with random entries D_{ii} on the diagonal such that

$$D_{ii} = \begin{cases} -1 & \text{with propability } \frac{1}{d} \\ 1 & \text{with probability } 1 - \frac{1}{d} \end{cases}, \quad \forall i \in [n].$$

1. Implement the matrix-vector product $\boldsymbol{\Phi}\boldsymbol{x}$ for $\boldsymbol{x} \in \mathbb{R}^n$ without explicitly building $\boldsymbol{\Phi}$ and without relying on any matrices.

   Hint: A clever trick might be used for a fast implementation.

2. Again, implement the "projected nearest neighbor" algorithm with this specific projection matrix using the matrix-vector product implemented in the previous question. Run it with $d = 300$.

3. Compute the theoretical time complexity of your algorithm with this particular matrix $\boldsymbol{\Phi}$ and compare it with numerical experiments.

4. How does the error evolve with respect to d? Like in the previous questions, plot how the accuracy evolves with respect to the ratio $\frac{d}{n}$. Compare your results with the one obtained with PCA in question 3.2.

3.5 Recommendations

Summarize the theoretical complexity results in a Table. Based on your numerical simulations and your theoretical analyses, which projection method would you recommend? Briefly, justify your choice.

4 Practical information

- *Group:* Make groups of 2 on the Moodle activity "Group choice for Project 1". Register as soon as possible.
- *Deadline:* **Wednesday 6th November 2024, 23:59**, in the "Project 1" activity on Moodle.
- *Material:* This assignment and one Jupyter Notebook to complete. The dataset should be downloaded via the link given above.
- *Deliverables:* Per group, one report in PDF format **.pdf** and one completed Jupyter Notebook in **.ipynb** format. If your Notebook has dependencies with external files (such as images or Python files **.py**), then the latter should be delivered too.
- *Report:*
  - 5 pages maximum, images included (4 pages should be enough), bibliography not included.
  - The font size should be 11 (single column) or 10 (double columns).
  - Answer all questions and subquestions.
  - Use Figures and Tables to communicate important results. Each Figure or Table should only answer one question or emphasize one aspect of your results.
  - We insist that you structure your report **in three sections**: a brief introduction recalling the context and the goal of the project, the main part containing your answers, and a brief conclusion including your recommendations of question 3.5 and a brief summary of your results. The sections should respectively represent around 10-75-15% of the document.
  - The report should be understandable for a person with the background of someone starting a master's degree as a civil engineer in applied mathematics.
- *Scientific approach:*
  - Theorems and results from the course involve assumptions, conditions, parameters, variables and constants. Be clear and rigorous when you use them.

  - Your experimental protocols should be cleverly designed, clearly explained and robust to noise. In particular, when randomness is involved, repeat your experiments a sufficient number of times to validate the consistency of your results.
  - A good analysis should always compare your theoretical expectations with your numerical results.

- *References:* You might be interested in looking up external resources (scientific articles, blogs, online implementations, ...). Cite them correctly and include a bibliography at the end of your report.

- *AI:* Using AI writing/coding assistant (such as Chat GPT) is allowed, but you have to add a section at the end of the report detailing in which clever way you used this tool.

- *Q&A:* Two Q&A sessions will take place on:

  - **Friday 25th October 2024, 10:45-12:45**, in BARB03 (instead of the exercise session).
  - **Thursday 31st October 2024, 15:00-16:00**, in the a.125 room on the first floor of the Euler building.

  If you come to a session, make sure to first explain in around 5 minutes what you've already done and to have prepared your questions in advance.

- *Contact:* As a last resort, you can send your questions and remarks directly to Bastien Massion (`bastien.massion@uclouvain.be`) and Nicolas Mil-Homens Cavaco (`nicolas.mil-homens@uclouvain.be`).